# Empirical Analysis of User Data in Game Software Development

The Story of Project Gotham Racing 4

Kenneth Hullett*
UC Santa Cruz
Santa Cruz, CA, USA
khullett@soe.ucsc.edu

Nachiappan Nagappan
Microsoft Research
Redmond, WA, USA

Eric Schuh
Microsoft Game Studios
Redmond, WA, USA

{nachin, eschuh}@microsoft.com

John Hopson
Bungie Studios
Bellevue, WA, USA
jhopson@bungie.com

## ABSTRACT

For several years empirical studies have spanned the spectrum of research from software productivity, quality, reliability, performance to human computer interaction. Analyses have involved software systems ranging from desktop software to telecommunication switching systems. But surprising there has been little work done on the emerging digital game industry, one of the fastest growing domains today. To the best of our knowledge, our work is one of the first empirical analysis of a large commercially successful game system. In this paper, we introduce an analysis of the significant user data generated in the gaming industry by using a successful game: Project Gotham Racing 4.

More specifically, due to the increasing ubiquity of constantly connected high-speed internet connections for game consoles, developers are able to collect extensive amounts of data about their games following release. The challenge now is to make sense of that data, and from it be able to make recommendations to developers. This paper presents an empirical case study analyzing the data collected from a released game over a three year period. The results of this analysis include a better understanding of the differences between long-term and short-term players, and the extent to which various options in the game are utilized. This led to recommendations for future development ways to reduce development costs and to keep new players engaged. A secondary goal for this paper is to introduce software game development as a topic of importance to the empirical software engineering community and discuss research results on a key difference area: data analytics on user data to customize user and development experiences.

## Categories and Subject Descriptors

D.2.8 [**Software**]: Metrics, K.8.0 [**Personal Computing**]: Games



## General Terms

Design, Measurement

## Keywords

Game design, Game development, Game metrics

## 1. EMPIRICAL RESEARCH AND GAMES

Empirical research in software engineering has typically focused on software systems ranging from the traditional telecommunication systems to more recent web services. There has been little research on the software engineering aspects of digital games (a.k.a. video games, computer games, electronic games, etc.; referred to simply as games for the remainder of this paper). Games are increasingly becoming an important part of the mainstream software development industry. PricewaterhouseCoopers (PwC) report *Global Entertainment and Media Outlook: 2007-2011* estimates that the video game market will increase from $31.6 Billion in 2006 to $48.9 Billion in 2011[1]. Games require significant software engineering effort and have become increasingly complex as games get more sophisticated [3]. Many of the issues in the development, production, and testing of games reflect those of the general software engineering community, and in many cases represent the state of the art. Research communities exist for specialized aspects of game development, such as SIGGRAPH's game track [15] for graphics or AAAI's Artificial Intelligence and Interactive Digital Entertainment for game AI [2]. There are already workshops in this regard that have been held co-located to the International Conference of Software Engineering [12]. That said, games are a significantly wide field and in this paper our goals are twofold:

- Identify a specific area of research and characterize its operation in the gaming community
- Investigate via data analytics the ability to improve game design

There are several differences between software development for games compared to software development for traditional software

* Kenneth Hullett was an intern at Microsoft Research when this work was performed.

[1]http://www.businessweek.com/innovate/content/aug2007/id20070813_120384.htm

systems. It is beyond the scope of this paper to assess these differences. In this paper we focus on one particular aspect, the importance of user testing. In recent years there has been a rise in interest in the collection and analysis of game metrics, and how they can be used to inform the game development process. As games have gotten larger and more complex, the need for such metrics to make sense of player behavior has increased. The number of reachable states in a modern commercial game title is enormous; without some way to simplify and represent collected data development teams would be unable to act on it in a timely matter.

Telemetry, or the collection of metrics, has become increasingly common in game development. As games have become more complex traditional playtesting is no longer able to provide sufficient coverage of all possible gameplay states or reveal potential emergent elements. Playtesting refers to the user testing wherein data is collected from players playing the game to identify defects and improve customer experience. This makes long-term metrics collection the only viable means to understand players and how they interact with the game.

But just collecting data is not enough, the data has to be distilled and interpreted before it can be used to inform development decisions. A detailed accounting of players' in-game actions is difficult to interpret even for a developer who is intimately familiar with the game. Simply knowing what a player did at a certain time means nothing without the context of what they did previously, what they did afterwards, and how that relates to the larger patterns of behavior throughout the game.

Previous academic work studied data on smaller scales in limited domains, and case studies from industry have shown ways various types of data can be used in to aid the development process. Our aim is to unite and advance these traditions by presenting a case study of analysis of large-scale data collected from *Project Gotham Racing 4,* a popular commercially released game.

This paper presents our case study by explaining the domain, our analyses, the conclusions we drew, and recommendations we were able to make. Some areas we explore include:

- Factors that hinder a player's advancement
- Differences between long-term and short-term players
- Differences between multi- and single player usage
- How players interact with the game in their first ten races and how this relates to long term behavior
- Utilization rates of various game play options and factors that contribute to them

This paper is organized as follows. In Section 2 we discuss the related work. In Section 3 we explain the various sources of data in games via quantitative (testing) and qualitative (subjective evaluations) aspects to characterize the process on how the game industry handles these activities. In Section 4 introduce Project Gotham Racing 4. In Section 5 we describe the data collected and Section 6 the data analysis and results. Section 7 presents the recommendations. We present conclusions in Section 8.

## 2. RELATED WORK

*A. Academic*

Presented here are some examples of academic work that explore data analysis in games.

Dixit and Youngblood performed user tests and used the data collected to create visualizations showing where users' attention was focused during gameplay [4, 5]. This information was then used to determine the best places to put relevant information to improve recall. The implications for game design are to give designers a better idea of where to place important clues and other information so that players are most likely to see it.

Kim et al. presented TRUE, a system for collection and visualization of data from user studies, and presented a case study of its use in Halo 2 [8], a popular First-Person Shooter (FPS) game. Their studies specifically looked for unintended difficulty increases introduced during development. Their tests collected data on player deaths and surveyed the subject's opinions on difficulty. They were able to identify several unbalanced elements in the game and correct them before release.

Weber and Mateas used data mining techniques on large amounts of collected data to understand player strategies in the game *StarCraft*. Over 5000 replays of expert matches were used as training data for a machine learning algorithm that predicted player strategies [20]. This predictor became a component of an AI bot that played *StarCraft* better than most other available techniques, thus helping to improve game AI.

Weber et al. used analysis of large datasets generated by players of the game *Madden NFL 11* to understand player retention [19]. They selected gameplay features to use as input to a machine learning algorithm that tried to predict how many games a player would play overall. The selected features were then studied to make recommendations that would increase player retention in the game.

Lewis and Wardrip-Fruin presented a case study of large-scale data collection and interpretation of *World of Warcraft* repositories for better understanding of player behavior [9]. They analyzed how long it took players from each class to reach level 80 (the highest level) in order to empirically evaluate whether the game design is balanced, and confirm or refute common folklore surrounding the game.

Miller and Crowcroft also studied *World of Warcraft*, but instead looked at player movement [10]. They analyzed several gameplay traces that utilized the same battleground to find patterns in player traffic. They were able to identify distinct patterns of player behavior, including patrollers, who moved between multiple points along standard routes, and guards who tended to remain within a small area. This analysis is useful to developers by showing how players are interacting with the game environment.

*B. Industry*

Articles in industry-focused publications like Gamasutra are a good source of information for ways in which data is used in the game industry. Some key examples are presented in this section.

Russell examined the combat design in *Uncharted 2* [13, 14]. They reflected on the previous game in the series and drew useful lessons, as well as described the process by which they iterated on the design of their current game. Levels were playtested repeatedly, with both telemetry and observational data being collected. This data was used to informed design changes and improve the game.

Adent discussed the development of *Forza Motorsport 3* and how testing and data analysis contributed [1]. A key factor for that team was having a stable, playable build at all times. This enabled constant iterative development of the game. Constant playability allowed for a constant stream of data for the designers to study and make changes accordingly.

Van der Heijden examined the usability testing done for *Swords & Soldiers* [18]. They describe the key questions the developers hoped to answer, the set up and testing process, and what they learned. In particular they were interested in improving the interface design and used eye-tracking data to see where players' attention was focused.

Another example of usability testing is in Thompson's article on *Halo 3* development [17]. They describe the extensive playtesting performed to improve the playability and balance of the game. Large numbers of players were observed and data was collected about how well they performed, leading designers to make adjustments. Players were also asked subjective questions about their level of enjoyment.

Another game in the Halo series, *Halo: Reach*, was subjected to a large beta test – over 2.7 million players and 16 million hours of testing [11, 16]. The result was not only finding and fixing bugs, but also significantly tweaking the gameplay by adjusting factors such as weapon damage, reload times, shield recharge rates, etc.

# 3. SOURCES OF DATA

*A. Internal Testing*

One of the earliest sources of data for game development teams is from their internal testing. This includes informal testing by the developers themselves and more formal testing by the QA team.

*1) Developers*

The earliest testers of a game are the development team itself, and therefore are the earliest creators of useful data about the game. In the early development, teams create small prototypes to test and explore new ideas. While these prototypes are generally discarded once the main development cycle begins, the lessons learned are an important initial source of data about what works and doesn't work in the game.

Once the game is fully in development, the team will continuously be testing the game. Of particular interest to designers is the play balance of the game. Level designers will play levels to ensure that they have the correct difficulty level for where they appear in the game. Matching increasing difficulty to the players' increasing skill as they learn the game is key to keeping players engaged.

*2) QA*

The main objective of the QA team is to find bugs and report them to the development team. Statistics from reported bugs are used to make production decisions in much the same way as they are used in traditional software development.

Many bugs are straightforward problems that the programmers, designers, and artists can easily address, but the QA team will often find problems with the playability of the game, including play balance issues. QA testers are often highly skilled game players, and continuously evaluate aspects of the game for difficulty, play time, and balance. Data collected from this playtesting can be used by the developers to make adjustments while the game is still in development.

*B. External Testing*

External testing is testing done by players from the community, rather than members of the development or QA teams. Releases of the game used for external testing are generally instrumented to collect data about the players' actions in the game.

*1) Usability Testing*

Usability testing is done with selected members of the target audience to better understand interactions with and reactions to the game. It is generally done under controlled psychological research protocols. To be effective, usability testing must be done late enough in the development cycle so that the game is representative of its final state, but not so late that it's costly to make changes.

In most cases, usability testing is the first time someone outside the organization plays the game. As the development and QA teams have been involved in the project for a long time, they are familiar with how the game is intended to be played and may not realize what is obvious or not to players. By putting a subject in a room and observing them play without instruction or interference, the development team can better gauge their expectations of how players will react to the finished game.

Typical outcomes of usability testing include the need for better tutorials to teach new players and clearer interfaces. Besides the qualitative assessment of players' reactions to the game, quantitative data about the players' specific actions can also be gathered.

*2) Beta Tests*

A beta test is a release of a nearly-complete version of a game to a limited set of players. Beta testers are generally selected from a pool of players of previous games.

In the past, beta tests consisted of sending copies of games to members of the pool, waiting for them to play, and receiving back questionnaire responses and comments. However, with the increasing ubiquity of internet connected game machines, the beta version can be downloaded directly to the tester's machine and play data can be reported directly to the development team.

Beta tests can also be contribute to the marketing of a game by giving players a preview of the game and building excitement about the release.

*3) Long-term Play Data*

While not actually testing per se, data gathered from players after a game's release can be an important source of data. Due to the increasing ubiquity of internet-connected game, development teams can easily collect player data indefinitely after release. If problems are found, teams can make changes and deliver a new version to players even after release.

Examples of useful data that can be obtained from long-term data are what achievements are earned, how quickly players progress, or favorite levels or game play modes. One well known example of long-term play data are the Halo heat maps published by Bungie Studios [17]. These show the locations of player deaths and kills by different weapons across all multiplayer maps. By examining these, the team can make adjustments for future releases.

Data from long-term play is particularly useful for maintaining play balance. A lack of balance may not have been appeared in earlier testing, but only becomes apparent after many months of play. An example would be an unanticipated dominant strategy. If, by observing play data, a team sees that a particular weapon has become favored, then they may want to adjust the balance to counter this.

Long-term data can also help teams plan the release of expansion content. When interest in a game starts to wane, developers can release new downloadable content that will entice players to continue playing. Also, examining at what point in their progress players start downloading new content can drive recommendation systems for future players.

*C. Subjective Evaluations*

*1) Surveys*

While much of the interest in game metrics is focused on quantitative data, qualitative data is also important. Survey data is generally collected along with the quantitative data collection during usability and beta testing. This data can be open ended, such as general questions about players' reactions to the game, or structured, such as rating various aspects of a game on a Likert scale.

*2) Reviews*

One source of expert data is reviews of games written by professional or non-professional journalists. The games industry is a large, international industry with hundreds of games released each year; game buyers consult reviews to determine what games are most worth spending their money on. By looking at reviews of their own and similar games, developers can decide what aspects to focus on to increase the likelihood of good reviews.

*3) Online Communities*

Gaming culture is increasingly involved and worldwide. Gamers don't play games in isolation; they comment upon and read other player's comments on various message boards and blogs dedicated to the subject.

Another aspect of online communities is expert players writing guides for new players. These guides, often called FAQs (from Frequently Asked Questions), are published at websites like GameFAQs.com [6]. Information found in FAQs includes complete walkthroughs of games, strategy guides, maps, and character creation guides.

By monitoring the online communities populated by their players, development teams can get a sense of how their game has been received by the gaming community and how their view of the game matches the design. If the walkthroughs miss some important aspect, then it was too hard to find. If the players' assessment of the strength and weaknesses of various elements don't match the team's expectations, then their play balancing may need adjustment.

*4) Post Mortems*

It is becoming increasingly common for industry-focused publications to publish game developers' post mortems after a game is released. This is a summary of what went right and wrong in the development process. By studying areas of development that were problematic in other projects, developers can better anticipate and avoid problems in their own projects.

## 4. PROJECT GOTHAM RACING 4

We present an analysis of long-term play data from a commercially released game. For this case study, we looked at data from *Project Gotham Racing 4* (PGR4), an Xbox 360 game developed by Bizarre Creations and published by Microsoft Game Studios in 2007 (Example screen shot – Figure 1).

PGR4 is an auto racing game and is representative of many games in the genre. Players have the option to play either single or multiplayer races organized into various game modes and event types. Game modes include, for example, career mode, a single player mode where the player earns money by competing in races, which in turn allows them to unlock other races and vehicles, leading to continuous advancement. Other game modes are multiplayer quick races, arcade mode, and time attack challenges. There are ten of these in total. Event types are the 29 specific challenges a player may compete in within a mode. These include things like street race, cone challenges, and elimination races.

The game features 134 vehicles, both cars and motorcycles, organized into 7 classes, A–G. The primary division between classes is performance, with A-Class being the highest performance and G-class being the lowest. Races are conducted on one of 121 routes spread out over 9 in-game locations. Locations are generally virtual representations of cities, such as Macau or Shanghai, while the routes are specific tracks laid out over the location.

**Figure 1: A screenshot from *Project Gotham Racing 4***

In the time since its release, PGR4 has been played extensively by its audience. Telemetry data was collected from players who opted in whenever they played while connected to the Xbox Live service, regardless of whether they were playing in multi- or single player races.

## 5. DATASET

Several datasets were collected from PGR4. The primary one analyzed was the Start of Race dataset. This contained approximately 3.1 million entries, one for each time a player started a race, including both multi- and single player races. Data about both the race and the player were logged, including:

- Type of event
- Route selected
- Vehicle selected
- Number of vehicles in race
- Player's career rating
- Number of previous events completed by player
- Total kudos earned by player

*A. Features*

For our analysis we looked at usage patterns for five game features of interest to the development team:

- Game modes
- Event types
- Routes
- Vehicles
- Vehicle classes

As these are the main options available to the player, patterns in their usage present a picture of how players are playing the game and what is most important to them.

*B. Subdivisions*

We felt it would be beneficial to separately examine players grouped according to their level of engagement with the game. To this end we subdivided the data into four groups based on the total number of races for that player in the entire dataset. The four groups were:

- Regular: > 200 races
- Mid 2: > 85 & ≤ 200
- Mid 1: > 13 & ≤ 85
- Infrequent: ≤ 13 races

For most analyses, we specifically compared the two most extreme groups: the regular and the infrequent players. This allowed us to make statements about how the behavior of the most enthusiastic players compared to the least engaged.

*C. Subsets*

In addition to studying the entire dataset, we examined three subsets for additional insight. We looked at multiplayer and single player races separately, and looked at the first ten races for each unique player. The motivation behind looking at the first ten races was to understand how a player's initial experience affects their subsequent engagement by the game. Differences that exist between infrequent and regular players in their first ten races may contribute to the likelihood that a new player will ultimately fall into one group or the other.

## 6. ANALYSIS AND RESULTS

We drew five conclusions from our examination of the Start of Race dataset:

- Regular players play more multiplayer races
- Regular players play more in career mode
- Many options (game modes, event types, routes, and vehicles) are underused
- A- & F-Class vehicles are most popular classes of vehicles
- C-Class vehicles equally or more popular than B-Class, especially among regular players

*A. Regular players play more multiplayer*

Within both the entire Start of Race dataset and the first ten races, regular players showed a clear preference for multiplayer game modes and event types.

For regular players, NETWORK_PLAYTIME was the 2nd most popular mode, used in 27.6% of races overall (see Figure 2). In contrast, for infrequent players, NETWORK_PLAYTIME is 3rd, at 16.1%, behind 2 single player modes (OFFLINE_CAREER at 47.0% and PGR_ARCADE at 19.6%) (see Figure 3).

In terms of event types, the most popular for regular players in the entire dataset was NET_STREET_RACE at 26.6%. For infrequent players, it was second at 10.5%, significantly less than the single player event type of STREET_RACE at 54.8%.

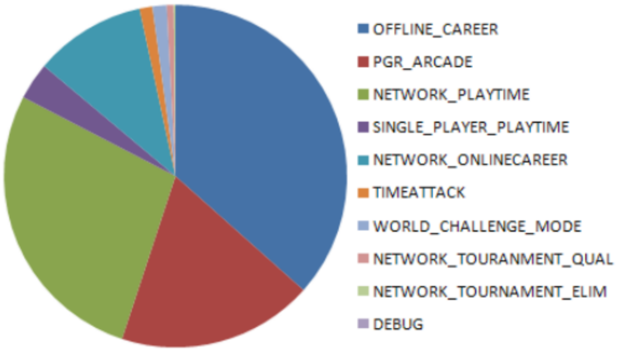


**Figure 2: Game Modes, Regular Players**

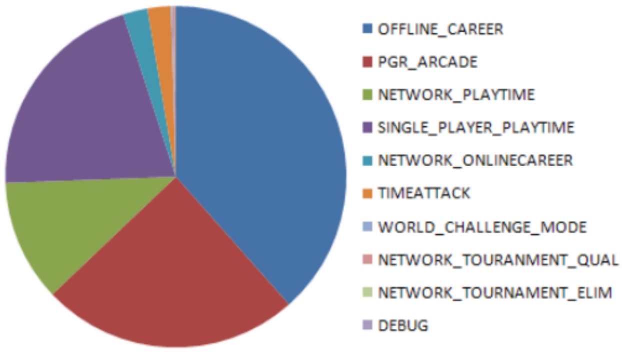


**Figure 3: Game Modes, Infrequent Players**

We see a similar pattern when looking at the first ten races only. 48% of races for regular player were in multiplayer game modes, compared to 20.8% for infrequent players. The most common multiplayer game mode, NETWORK_PLAYTIME, was significantly more preferred by regular players, 35.5% versus 7.6% for infrequent players.

For regular players NET_STREET_RACE was the most popular event type by an overwhelming margin: 39% of races, with single player STREET_RACE a distant second at 15.5%. For infrequent players, the single player event types of STREET_RACE and TIMEATTACK were vastly more preferred (24.9% and 20.8% respectively) over NET_STREE T_RACE (3rd at 9.4%).

- Regular players used the NETWORK_PLAYTIME game mode more than infrequent players
- Regular players used the NET_STREET_RACE event type more than infrequent players
- In their first 10 races, regular players used the NETWORK_PLAYTIME game mode more than infrequent players by a large margin
- In their first 10 races, regular players used the NET_STREET_RACE event type more than infrequent players by a large margin

*B. Regular players play more career mode*

When regular players do play single player races, they are more likely to do so in career mode than infrequent players.

In the entire dataset, OFFLINE_CAREER was the most popular game mode overall for regular players: 36.6%, followed by the aforementioned multiplayer mode NETWORK_PLAYTIME at 27.6% (see Figure 2). In contrast, the non-career modes of SINGLE_PLAYER_PLAYTIME and PGR_ARCADE were more preferred by infrequent players (20.6% v. 3.5% and 24.5% v. 18.46% respectively) (see Figure 3).

When looking at data from the single player races only, OFFLINE_CAREER was the most popular for both regular and infrequent players. 59.9% of single player races for regular players were in OFFLINE_CAREER and 47.5% for infrequent players. This may not seem like a large difference, but when we look at the primary non-career mode, SINGLE_PLAYER_PLAYTIME, the difference becomes more apparent. Regular players used SINGLE_PLAYER_PLAYTIME in only 5.8% of single player races, while infrequent players used it in 24.2%.

We see a difference in the first ten races as well. Regular players prefer OFFLINE_CAREER career more than infrequent players (36.5% v. 22.2%). By contrast, infrequent players were more likely than regular players to play non-career modes TIMEATTACK (20.1% v. 0.5%) and PGR_ARCADE (26.4% v. 6.5%).

- OFFLINE_CAREER was the most popular game mode among regular players
- SINGLE_PLAYER_PLAYTIME was used more by infrequent players overall and in their first ten races
- Regular players used OFFLINE_CAREER more in their first ten races

*C. Many options were underused*

Our analysis showed that large amounts of the options available in the game were used in so few instances that they could have been removed from the game entirely. In four of the features we examined, 20% to over 70% of available options were used in less than 1% of races. This suggests that savings in development times and costs could be realized in future games by offering fewer options without negatively affecting the players' overall experience. When looking at the entire dataset,

- 22% (2 of 9) game modes,
- 41% (12 of 29) event types,
- 67% (81 of 121) routes,
- and 78% (104 of 134) vehicles

were used in less than 1% of races each.

*1) Game Modes*

As shown in Table 1, OFFLINE_CAREER (a single player mode) was the most commonly used game mode by far, with NETWORK_TOURANMENT_QUAL and NETWORK_TOURNAMENT_ELIM being used in less than 0.5% of races. If fact, the 7 least used modes account for only 15% of races overall.

**Table 1. Game Modes**

| Game Mode | Races | % of Total |
|---|---|---|
| OFFLINE_CAREER | 1479586 | 47.63% |
| PGR_ARCADE | 566705 | 18.24% |
| NETWORK_PLAYTIME | 584201 | 18.81% |
| NETWORK_ONLINECAREER | 193091 | 6.22% |
| SINGLE_PLAYER_PLAYTIME | 185415 | 5.97% |
| TIMEATTACK | 43942 | 1.41% |
| WORLD_CHALLENGE_MODE | 36581 | 1.18% |
| NETWORK_TOURANMENT_QUAL | 13847 | 0.45% |
| NETWORK_TOURNAMENT_ELIM | 2713 | 0.09% |

When we look at just multiplayer game modes we see an even larger disparity: the top two modes account for 98% of all multiplayer races.

*2) Event Types*

When looking at event types, we again see a rapid drop off in popularity with the least popular types receiving only trivial usage. A reduced version of this data is shown in Table 2.

**Table 2. Event Types (Reduced)**

| Group | Races | % of Total |
|---|---|---|
| STREET_RACE | 795334 | 25.60% |
| NET_STREET_RACE | 543491 | 17.50% |
| ELIMINATION | 216042 | 6.95% |

| | | |
|---|---|---|
| HOTLAP | 195949 | 6.31% |
| … | | |
| TESTTRACK_TIME | 7484 | 0.24% |
| NET_CAT_AND_MOUSE_FREE_ROAM | 3989 | 0.13% |
| CAT_AND_MOUSE | 53 | 0.00% |

Single player street races were the most popular event type, followed by multiplayer street races and elimination races (knock out stages in tournaments), whereas 12 of the 29 event types were used in less than 1% of races. The underutilization of event types is even more pronounced when looking at multiplayer races only (7 of 16 event types used in less than 0.1% of races).

*3) Routes*

While 67% of the available routes were used in less than 1% of races each, collectively they account for 36% of races. i.e., two-thirds of races occur on one-third of the routes. Developers would likely not support a proposal to eliminate such a large portion of potential gameplay, so we looked at even smaller percentages of use and found that

- 47 (39%) were used in less than 0.5%,
- 19 (16%) were used in less than 0.25%,
- and 8 (7%) were used in less than 0.1%

of total races.

The 47 routes used in less than 0.5% of races account for 13% of overall usage, a much more palatable percentage to consider removing, while still leaving 70+ routes available for players.

*4) Vehicles*

Similarly with routes, a wide variety of vehicles adds to depth of gameplay even if a significant portion is rarely used. Furthermore, the number of available vehicles in a driving game can be an important point in the marketing strategy.

The 104 of the 134 vehicles that are used in less than 1% of races each collectively represent 38% of usage. Furthermore,

- 72 (54%) were used in less than 0.5%;
- 50 (37%) were used in less than 0.25%,
- and 12 (9%) were used in less than 0.1% of total races.

The 50 vehicles used in less than 0.25% of races each represent less than 7% of the total races.

*5) Vehicle Classes*

We can also look at vehicles in terms of their classes. The vehicles in the game are grouped into 7 classes based on performance. As seen in Table 3, A-Class vehicles were used nearly twice as often as the next most popular class, while Classes B though F were close in popularity, ranging from 10-15% of all races.

**Table 3. Vehicle Class**

| Vehicle Class | Races | % of Total |
|---|---|---|
| A_Class | 908581 | 29.25% |
| F_Class | 478944 | 15.42% |
| C_Class | 465889 | 15.00% |
| B_Class | 454594 | 14.63% |
| D_Class | 386862 | 12.45% |
| E_Class | 338938 | 10.91% |
| G_Class | 69625 | 2.24% |

Also, G-Class was considerably less popular, being used in about 2% of races overall. This suggests that the number of classes can be reduced. Players have little interest in the low-performance G-Class, and perceive little difference between the other classes except A-Class.

*D. A- & F-Class vehicles most popular*

As seen in Table 3, A-Class vehicles were the most popular by a considerable margin. They were also the most preferred in multiplayer (53.6%) and in the first ten races (32.5%). These are the highest performance vehicles, so we would expect them to be most preferred by regular players, and they were (36.2% v. 20.2% for infrequent players). However, they were still used significantly by infrequent players, being the second most popular in the first ten races (33%) and overall (20.2%) and most popular in multiplayer races (54.5%).

While in the overall dataset F-Class doesn't appear significantly more popular than B- through E-Classes, when we look at subsets of the data we see certain trends. Amongst infrequent players, F-Class was by far the most popular, 55.4% overall (see Figure 4) and 47% in single player races only. This seems like an obvious result as F-Class vehicles are the only ones initially available in career mode at the start of the game, but as shown above, infrequent players are less likely to play in career mode.

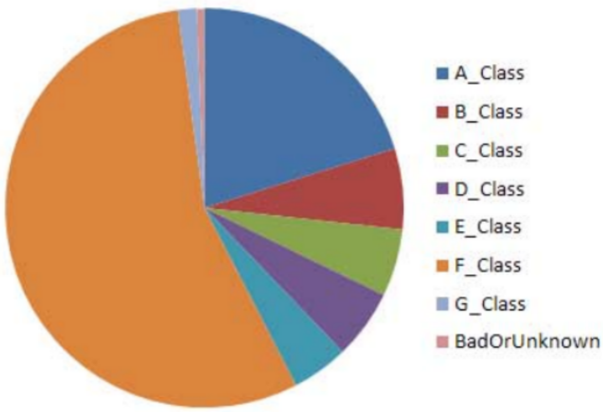


**Figure 4: Vehicle Class, Infrequent Players**

- A-Class vehicles were the most often used overall, in multiplayer, and in players' first ten races
- A-Class vehicle were the second most often used by infrequent players overall and in their first ten races
- F-Class vehicles were the most often by infrequent players

*E. C-Class & B-Class equally popular*

As seen in Table 3, C-Class was slightly more popular than B-Class overall. This may not seem significant, but when we look at the data for the first ten races, we see that C-Class was the second most popular class among regular players at 26% v. 13% for B-Class. This suggests that C-Class cars have characteristics that make them more appealing to players than the higher-performance B-Class vehicles.

- In their first 10 races, regular players used C-Class vehicles twice as often as B-Class vehicles

## 7. RECOMMENDATIONS

The five conclusions we reached after examining the Start of Race dataset led to 4 recommendations for future development that would be applicable to many different games in the racing game genre, and could possibly be generalized to other games:

- New players should be encouraged to play in career mode
- New players should be encouraged to use F-Class vehicles in multiplayer
- Development time and costs could be reduced by having fewer available options
- Reduce the number of vehicle classes from 7 to 5

*A. New players should be encouraged to play career mode*

As discussed above, regular players are more likely to play in career mode, both overall and in the first ten races. This suggests that playing in career mode increases the likelihood that a player will continue playing the game for a much longer time. Players enjoy progression, and being presented with a series of increasing challenges and rewards, such as advancing through the stages in career mode, will cause them to be engaged and keep playing.

The data suggests that many new players come into the game, experiment with various game modes and event types in their first few races, and then stop playing. If they could be drawn into the challenge/reward structure of career mode they would be more likely to continue playing. The early career races are designed to be easy, so most players will start winning early, unlocking more cars and routes that they are then eager to try out.

*B. New players should be encouraged to use F-Class vehicles in multiplayer*

While infrequent players were shown to prefer F-Class vehicles in single player races, they had as high a preference for A-Class vehicles in multiplayer as regular players. Given that the learning curve for A-Class vehicles is quite steep, this may be a factor in infrequent players losing interest in the game. If, in one of their earliest experiences with the game, a player joins a multiplayer race with experienced players on a track they are unfamiliar with, picks one of the fastest cars available, and then crashes in the first turn, they are likely to become frustrated and stop playing.

Alternatively, new players could, by default, be sent to multiplayer races only with other new players, specifically on tracks that are available early in the single player game. The only vehicles available would be the F-Class vehicles, so they wouldn't feel compelled to select an A-Class vehicle merely to be competitive with other players. These initial experiences in multiplayer would be gentler, on tracks and using vehicles they are familiar with, and against other players of similar skill levels.

*C. Development time and costs could be reduced by having available options*

Our analysis showed that 20-70% of the available options were used in less than 1% of races each. As asset creation is a major expense in game development, reducing little-used options could significantly reduce costs and development time while having little impact on players' experience. Each vehicle in the game, for example, represents a significant investment: a 3d artist must model it, a texture artist must decorate it, a designer must to tweak its performance values, and testers must rigorously use it in a variety of conditions to make sure there are no problems. Creating new routes requires artists, designers, and testers, while new event types require engineering effort.

That being said, there are benefits to having little used content available in the game. It can extend the life of a game for players; they can explore rarely-used options when they grow tired of the game. A wide variety of options can lead to emergent play as players find uses for content that developers never anticipated. The amount of content can be useful in the marketing of a game; being able to say that you has more vehicles or event types than your competitors can drive sales.

While excessively pruning available content in future games might not be preferred, a reduction of 20% across the board could reduce costs and development times significantly while the back of the box could still boast that the game contains more than 100 vehicles.

*D. Reduce number of vehicle classes from 7 to 5*

In addition to reducing the sheer amount of content, removing complexity from the game can reduce cognitive overhead for the player. In particular, the 7 vehicle classes are an unnecessary element that does not enhance the game experience for the player.

The analysis showed that G-Class vehicles were used in about 2% of races overall. These are mostly low-performance specialty and historic vehicles that are not generally of much use to players throughout the game. Any that developers feel are important enough to keep could be moved into other classes.

The analysis also showed little difference in preference for Classes B−E. While having stages of progression is important to the learning curve, fewer steps would achieve the same effect. In particular, C-Class is preferred over B-Class in some instances, suggesting there's little difference and the two could be combined.

The resulting 5 classes should offer sufficient ramp-up in difficulty for the player to progress though the game without any sudden increases.

## 8. CONCLUSIONS

This paper presented a series of analyses performed on data collected on players in the game *Project Gotham Racing 4*. We looked for patterns within large data sets that provide insight into player behavior. We learned there were key differences in how regular and infrequent players approached the game, and how what players do in their earliest exposure to the game can affect their desire to continue playing. We also found that much of the available options for gameplay were rarely used by players.

From the patterns in the data we made recommendations for future development. Many rarely utilized options could be removed with no negative impact on players. A more structured introduction would keep new players engaged and increase the likelihood that they will continue playing.

These conclusions could be applied generally to a wide variety of games in different genres. Developers who are inclined to add many options to their games should consider the result that players in PGR4 tended to focus mainly on the game's core features. Providing tutorials and a gentle early difficulty curve can help ease new players into a game and keep them playing. This paper also shows that simple analysis to begin with can greatly help advance the state of the art in software engineering in the games domain.

It is also important to call out several points.

a. Telemetry, or the collection of metrics, has become increasingly common in game development. The contribution of this paper is to show how data analysis, even exploratory, on data from games can have potentially far-reaching software engineering implications in the empirical community. There is the potential to evaluate how these new classes of software systems work with various empirical processes and practices.

b. The paper necessarily spends a lot of time explaining the domain of the analysis to provide context to the reader. A secondary goal of this paper is to introduce this topic of software engineering research for games to the broader software engineering community and expose the potential for research in games. We have presented the various sources of data available in game analytics. There are several open questions in the software engineering research for games community, ranging from the requirements engineering: how are requirements for games defined, where the major emphasis is based on user interaction and real-time feedback, the use of personas for requirements documents etc.; to testing and analysis: how can games be tested in the lab, simulating user behavior. We hope in coming years the software engineering community as a whole will embrace the games domain to investigate and address the important software engineering challenges facing games.

c. The paper is specific to Project Gotham Racing 4, though techniques similar to those outlined in this paper could be applied to any game dataset. As this case study involves an already released game, we were limited to making suggestions for future releases, but similar techniques could be applied to beta test data for a game in development. The larger the dataset, the more pronounced the patterns will be, but conclusions drawn from a pool of beta testers representative of the target audience are likely to be more insightful then those that can be gained through traditional playtesting. Similar analysis could be performed for different game genres, ranging from FPS (First Person Shooter) or strategy games to educational and physical activity games.

## ACKNOWLEDGEMENTS

We would like to gratefully acknowledge the participants of ISERN in the special session on Games in 2009. Their encouragement was instrumental in submitting this paper to ESEM. Part of this paper, specifically Section 3, appeared as part of an ICSE NIER 2011 paper [7]. The rest of the paper has significantly newer results. We would like to thank the ICSE NIER reviewers who provided much encouragement and support which caused us to submit this paper. We would like to thank all the users who played PGR4, the data collection team in Xbox, Thomas Zimmermann, Christian Bird, Tom Ball, Jim Whitehead, and the broader ESEM and ISERN communities which have encouraged the publishing of research on Games.

## REFERENCES

[1] Adent, D. *Forza Motorsport 3* and Predictable Development. Gamasutra.com. http://www.gamasutra.com/view/feature/6182/forza_motorsport_3_and_predictable_.php

[2] Annual Conference on Artificial Intelligence and Interactive Digital Entertainment. http://www.aiide.org/

[3] Blow, J. Game Development: Harder Than You Think. *Queue* 1, 10 (February 2004), 28-37, 2004.

[4] Dixit, P. N. and Youngblood, G. M. Optimal information placement in an interactive 3D environment. In *Proceedings of the 2007 ACM SIGGRAPH symposium on Video games*. 2007.

[5] Dixit, P. N. and Youngblood, G. M. Understanding information observation in interactive 3D environments. In *Proceedings of the 2008 ACM SIGGRAPH symposium on Video games*. 2008.

[6] GameFaqs. http://www.gamefaqs.com/

[7] Hullett, K., Nagappan, N., Schuh, E., Hopson, J. Data analytics for game development: NIER track. In *Proceedings of the International Conference on Software Engineering*. 2011.

[8] Kim, J., Gunn, D., Schuh, E., Phillips, B., Pagulayan, R., and Wixon, D. Tracking real-time user experience (TRUE): a comprehensive instrumentation solution for complex systems. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*. 2008.

[9] Lewis, C. and Wardrip-Fruin, N. Mining game statistics from web services: a World of Warcraft armory case study. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*. 2010.

[10] Miller, J. L. and Crowcroft, J. Avatar movement in World of Warcraft battlegrounds. In *Proceedings of the 8th Annual Workshop on Network and Systems Support for Games* (NetGames '09). 2009.

[11] Nutt, C. *Halo: Reach* - The Beta Story. Gamasutra.com. http://www.gamasutra.com/view/feature/5875/halo_reach__the_beta_story.php

[12] Proceedings of the 1st International Workshop on Games and Software Engineering, ISBN: 978-1-4503-0578-5, New York, NY. http://dl.acm.org/citation.cfm?id=1984674&picked=prox

[13] Russell, B. A Deeper. Look into The Combat Design Of *Uncharted 2*. Gamasutra.com. http://www.gamasutra.com/view/feature/5883/a_deeper_look_into_the_combat_.php

[14] Russell, B. Designing Combat Encounters In *Uncharted 2*. Gamasutra.com. http://www.gamasutra.com/view/feature/5945/designing_combat_encounters_in_.php

[15] SIGGRAPH, Games Papers 2011. http://www.siggraph.org/s2011/for_submitters/game-papers

[16] Sofge, E. How *Halo: Reach* Was Created: Insider's Guide. PopularMechanics.com. http://www.popularmechanics.com/technology/gadgets/video-games/how-halo-reach-was-created-insiders-guide

[17] Thompson, C. *Halo 3*: How Microsoft Labs Invented a New Science of Play. Wired.com. http://www.wired.com/gaming/virtualworlds/magazine/15-09/ff_halo

[18] Van der Heijden, J. Successful Playtesting In *Swords & Soldiers*. Gamasutra.com http://www.gamasutra.com/view/feature/5939/successful_playtesting_in_swords__.php

[19] Weber, B., John, M., Mateas, M., Jhala, A. Modeling Player Retention in Madden NFL 11. In Proceedings of *Innovative Applications of Artificial Intelligence*. 2011.

[20] Weber, B., Mateas, M. A data mining approach to strategy prediction. In *IEEE Symposium on Computational Intelligence and Games, 2009.*